package com.twitter.simclusters\_v2.scalding.topic\_recommendations.model\_based\_topic\_recommendations

import com.twitter.ml.api.util.FDsl.\_

import com.twitter.ml.api.{DataRecord, FeatureContext, IRecordOneToOneAdapter}

case class UserTopicTrainingSample(

userId: Long,

followedTopics: Set[Long],

notInterestedTopics: Set[Long],

userCountry: String,

userLanguage: String,

targetTopicId: Int,

userInterestedInSimClusters: Map[Int, Double],

followedTopicsSimClusters: Map[Int, Double],

notInterestedTopicsSimClusters: Map[Int, Double])

class UserTopicDataRecordAdapter extends IRecordOneToOneAdapter[UserTopicTrainingSample] {

import UserFeatures.\_

/\*\*

\* Get its feature context used to annotate the data.

\*

\* @return feature context

\*/

override def getFeatureContext: FeatureContext = UserFeatures.FeatureContext

/\*\*

\* Adapt record of type T to DataRecord.

\*

\* @param record raw record of type T

\*

\* @return a DataRecord

\*

\* @throws com.twitter.ml.api.InvalidFeatureException

\*/

override def adaptToDataRecord(record: UserTopicTrainingSample): DataRecord = {

val dr = new DataRecord()

dr.setFeatureValue(UserIdFeature, record.userId)

dr.setFeatureValue(

UserSimClusterFeatures,

record.userInterestedInSimClusters.map {

case (id, score) => id.toString -> score

})

dr.setFeatureValue(FollowedTopicIdFeatures, record.followedTopics.map(\_.toString))

dr.setFeatureValue(NotInterestedTopicIdFeatures, record.notInterestedTopics.map(\_.toString))

dr.setFeatureValue(UserCountryFeature, record.userCountry)

dr.setFeatureValue(UserLanguageFeature, record.userLanguage)

dr.setFeatureValue(

FollowedTopicSimClusterAvgFeatures,

record.followedTopicsSimClusters.map {

case (id, score) => id.toString -> score

})

dr.setFeatureValue(

NotInterestedTopicSimClusterAvgFeatures,

record.notInterestedTopicsSimClusters.map {

case (id, score) => id.toString -> score

})

dr.setFeatureValue(TargetTopicIdFeatures, record.targetTopicId.toLong)

dr.getRecord

}

}